Understanding Diversity based Pruning of Neural Networks — Statistical Mechanical Analysis

Rupam Acharyya[1], Boyu Zhang*[1], Ankani Chattoraj*[1], [1] Shouman Das[1], Daniel Štefankovič [1]

University of Rochester[1]
racharyy@cs.rochester.edu, bzhang25@u.rochester.edu, achattor@ur.rochester.edu, sdas13@ur.rochester.edu, stefanko@cs.rochester.edu

Abstract

Deep learning architectures with a huge number of parameters are often compressed using *pruning* techniques to ensure computational efficiency of inference during deployment. Despite multitude of empirical advances, there is no theoretical understanding of the effectiveness of different pruning methods. We address this issue by setting up the problem in the statistical mechanics formulation of a teacher-student framework and deriving generalization error (GE) bounds of specific pruning methods. This theoretical premise allows comparison between pruning methods and we use it to investigate compression of neural networks via *diversity-based pruning* methods. A recent work showed that *Determinantal Point Process* (DPP) based *node* pruning method is notably superior to competing approaches when tested on real datasets. Using GE bounds in the aforementioned setup we provide theoretical guarantees for their empirical observations. Another consistent finding in literature is that sparse neural networks (*edge pruned*) generalize better than dense neural networks (*node pruned*) for a fixed number of parameters. We use our theoretical setup to prove that baseline *random edge pruning* method performs better than *DPP node pruning* method. Finally, we draw motivation from our theoretical results to propose a *DPP edge pruning* technique for neural networks which empirically outperforms other competing pruning methods on real datasets.

1 Introduction

Deep neural networks have achieved impressive results in a wide variety of applications such as classification [19,26], image processing [3,25], natural language processing [6,7,36], etc. Most of these networks use millions and sometimes even billions of parameters which makes inference computationally expensive and memory intensive [7]. To address this, researchers explore pruning techniques with the primary goal of comparing performance on real datasets without any theoretical guarantees. In this work, we take a step to bridge the gap between empirical observations from pruning studies and their theoretical justification. We compare the quality of different pruned feedforward neural networks under the *teacher-student* framework [10,32–34] in the thermodynamic limit (input dimension goes to infinity) using *generalization error bounds*, a theoretical measure of performance of machine learning models on unseen test data [37] .
Amongst various pruning techniques, we focus on the class of methods where pruned networks do not undergo retraining. A fairly recent work in this category [29] empirically investigates a node pruning technique where a diverse subset of nodes are preserved in a given layer using DPP [20, 27]. In the first part of this paper we provide theoretical guarantees for their empirical observations.
[4] recently reviewed empirical results across various pruning studies to conclude that sparse models obtained after edge/connection (used interchangeably) pruning outperforms dense ones obtained after node pruning for a fixed number of parameters. We extend the theoretical setup and compare node and edge pruning techniques which are within the scope of our investigation, to provide a theoretical justification of their empirical observation driven claim.

*equal contribution

Research advances in neurophysiology have shown the existence of diverse synapses in the brain and also established that unused synapses are eliminated to conserve energy and ensure efficient information transmission [5, 30, 31]. Drawing motivation from the brain and our theoretical insights, we extend work by [29] to investigate empirical potentials of pruning that focus on synaptic (weight/connection/edge) diversity in feedforward neural networks via DPP. We emphasize that DPP edge pruning technique is complementary to prior network compression techniques and focuses on theory motivated implementation of 'diversity based synaptic pruning'.
Our main contributions are: (1) We use GE bounds on the teacher-student framework to compare different pruning methods, which to the best of our knowledge, is the first theoretical advance in understanding pruning methods. (2) We prove that DPP node pruning outperforms random and importance node pruning methods, previously shown by [29] empirically. (3) We also theoretically show that random edge pruning performs better than DPP node pruning which is consistent with empirical observations from pruning literature that sparse models outperform dense models [4]. (4) Finally, we propose DPP edge pruning method for feedforward networks which uses synaptic diversity to remove redundant edges and outperforms competing methods on real data. We also observe that DPP edge pruning finds a sparse network which performs better than the unpruned dense network, hence loosely connecting it to the Lottery Ticket Hypothesis [8].

2 Related Work

Broadly, two types of pruning techniques are used to compress neural networks: *unstructured pruning* and *structured pruning*. Seminal studies on unstructured pruning [13, 22] remove unimportant network weights based on the Hessian of the network's error function. Among others, alternative approaches include low rank matrix factorization of the final weight layers [35] or pruning the unimportant connections below a threshold [12]. Studies under structured pruning regime remove entire neurons/nodes (used interchangeably henceforth) keeping the networks dense [14,15,24]. Though dense networks can benefit from modern hardware, sparse models outperform dense ones for a fixed number of parameters across domains [11,17,23]. [4] is a recent review which highlights these observations after investigating 81 studies on pruning techniques. Our work is closely relates to [29], where a DPP sampling technique is used to select a set of diverse neurons/nodes to be preserved during pruning. The authors also introduce a *reweighting* procedure to compensate contributions of the pruned neurons in the network. Finally, they compare DIVNET (DPP node pruning with reweighting as in [29]) with random and importance node pruning [14] on real datasets.
Providing theoretical guarantees for neural networks is a known challenge. Pioneering work by [32–34] analyzes the GE dynamics form the statistical mechanics perspective on *teacher-student* framework [9] to understand the performance of neural networks on unseen test data. All our theoretical analyses throughout this work closely follow [1, 10], who analyzed results for the case where the student network has more number of hidden nodes than the teacher network.

3 Preliminaries

Determinantal Point Process (DPP): DPP [27] is a probability distribution over power set of a ground set $\mathcal{G}$, here finite. DPP is a special case of negatively associated distributions [16] which assigns higher probability mass on diverse subsets. Formally, a DPP with a marginal kernel L ($\in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$) is: $\mathbb{P}[\mathbf{Y} = Y] = \frac{\det(L_Y)}{\det(L+I)}$, where $Y \subseteq \mathcal{G}$ and L_Y is the principal submatrix defined by the indices of Y. We use k-DPP to denote the probability distribution over subsets of fixed size k.
DPP node pruning: [29] uses DPP to propose a novel node pruning method for feed forward neural network. They define information at node i of layer l as $\boldsymbol{a}_i^l (= (a_{i1}^l, \ldots, a_{in}^l))$, where a_{ij}^l is the activity of node i of layer l on j^{th} input. Here $\boldsymbol{a}_i^l = g\left(\boldsymbol{b}_i^l\right)$, where $\boldsymbol{b}_i^l = \sum_{i=1}^{n_{l-1}} w_{ij}^{l-1} \boldsymbol{a}_i^{l-1}$ is the information at node i of layer l before activation. A layer is pruned by choosing a subset of hidden nodes using a DPP kernel: $\boldsymbol{L}$ $(= \boldsymbol{L'} + \epsilon \boldsymbol{I})$, where, $\boldsymbol{L'}_{st} = \exp(-\beta \left\|\boldsymbol{a}_s^l - \boldsymbol{a}_t^l\right\|^2)$ and β is a bandwidth parameter. The matrix $\boldsymbol{L}$ is of dimension $n_l \times n_l$, as total number of nodes in layer l is n_l. By the property of DPP, this procedure will keep a diverse subset of nodes for each layer w.r.t. information obtained from the training data. A *reweighting* technique (see Section 2.2 of [29]) is then applied to outgoing edges of retained nodes to compensate for information lost in that layer

Table 1: Notations used in Theorems

Notations	Explanations	Notations	Explanations	Notations	Explanations
n	number of inputs	N	dimension of the input	n_l	number of nodes in layer l
v_i^l	i^{th} node in layer l $(1 \leq i \leq n_l)$	a_{ij}^l	activation of v_i^l on j^{th} input	M	number of teacher hidden nodes
e_{ij}^l	edge from v_i^l to v_j^{l+1} $(1 \leq i \leq n_l$ and $1 \leq j \leq n_{l+1})$	w_{ij}^l	weight of e_{ij}^l $(1 \leq i \leq n_l$ and $1 \leq j \leq n_{l+1})$	K	number of student hidden nodes
k_n	number of student hidden nodes kept after node pruning	k_e	number of incoming edges of a hidden node kept after edge pruning	v^*	second layer weight of teacher network

due to node removal. Note that DIVNET denotes DPP node pruning with reweighting as in [29].

Online Learning in Teacher-Student Setup [10]: We use a two-layer perceptron which has N input units, M hidden units and 1 output unit as the *teacher network* to generate labels for i.i.d Gaussian input, $\boldsymbol{x}^t = (x_1^t, \ldots, x_N^t)$ $(x_i^t \in \mathcal{N}(0,1)\ \forall i \in \{1, \ldots, N\})$. Let $\theta^* = \{\boldsymbol{w}^*(\in \mathbb{R}^{M \times N}), v^* \in \mathbb{R}^M\}$ denote the fixed parameters of the teacher network. The label y^t of the input $\boldsymbol{x}^t$ $(t = 1, 2, \ldots)$ is given as,

$$y^t = \sum_{m=1}^{M} v_m^* g\left(\frac{w_m^* \boldsymbol{x}^t}{\sqrt{N}}\right) + \sigma \zeta^t, \tag{1}$$

where $\zeta^t \sim \mathcal{N}(0,1)$ is the output noise, and g is the sigmoid activation function. The input and teacher generated labels $(\{(\boldsymbol{x}^1, y^1), \ldots\})$ are used to train a two-layer *student network* with N input units, K hidden units $(K \geq M)$ and 1 output unit using online SGD learning method. We consider the quadratic training loss, i.e.,

$$L(\theta) = \frac{1}{2}\left[\sum_{k=1}^{K} v_k g\left(\frac{w_k \boldsymbol{x}^t}{\sqrt{N}}\right) - y^t\right]^2, \tag{2}$$

where $\theta = \{\boldsymbol{w}, \boldsymbol{v}\}$ denotes the parameter of the student network. [10] showed that GE $\epsilon(f)$ (expected error on the unseen data, for details see S31 of [10]) for the student network is a function of the following *order parameters*,

$$Q_{ik} = \frac{w_i^T w_k}{N}, \quad R_{in} = \frac{w_i^T w_n^*}{N}, \quad R_{mn} = \frac{w_m^{*T} w_n^*}{N}. \tag{3}$$

Intuitively, these order parameters measure the similarities between and within the hidden nodes of teacher and student networks. Our theoretical results assume [10]:

(A1) If $\boldsymbol{x} = (x_1, \ldots, x_N)$ is an input then $x_i \in \mathcal{N}(0,1)$. Also, $N \to \infty$.
(A2) Both the teacher and the student networks have only one hidden layer.
(A3) $K \geq M$ and $K = Z \cdot M$ where $Z \in \mathbb{Z}^+$.
(A4) The activation in the hidden layer is sigmoidal for both teacher and student network.
(A5) The output $\in \mathbb{R}$ (i.e., regression problem).
(A6) The order parameters (see section 3) satisfy the ansatz as in (S58) - (S60) of [10].
(A7) No noise is added to the labels generated by the teacher network, i.e., $\sigma = 0$ in (1).

4 GE of Pruned Network in Teacher-Student Setup

We compare the performance of student networks pruned using different techniques as in Table 2 by analyzing their GE (see Figure 1). For node and edge pruning comparison, we choose the parameters k_n and k_e (see Table 1) such that the total number of parameters of the networks remain same, i.e., they satisfy,

$$\frac{k_n}{K} = \lim_{N \to \infty} \frac{k_e}{N} = c, \tag{4}$$

where $c \in [0,1]$ is a constant. It is important to note that since we assume that the number of student nodes is more than the number of teacher nodes, which means multiple student nodes learn the same teacher node (see Figure 3 of [10]; also in Figure 1: two student hidden nodes learn one teacher hidden node, shown in same color). From [10], we know that, in noiseless case ($\sigma = 0$ in (1)), the student network learns the teacher network completely when trained till convergence, i.e., the GE becomes 0. When we prune the student network, this GE increases, which we then analyze for different types of pruning under certain assumptions (see Section 3 (A1)-(A7)).

Table 2: Different pruning methods and notations for their GE. Here f denotes the pruned student network. u.a.r. and w.p. stand for *uniformly at random* and *with probability* resepectively.

Pruning Method	**Procedure**	**Retained Parameters**	**GE without reweighting**	**GE with reweighting**
Random Node	Keep k_n nodes u.a.r.	k_n hidden nodes	$\epsilon_{k_n}^{Rand\ Node}(f)$	$\hat{\epsilon}_{k_n}^{Rand\ Node}(f)$
Importance Node	[14]	k_n hidden nodes	$\epsilon_{k_n}^{Imp\ Node}(f)$	$\hat{\epsilon}_{k_n}^{Imp\ Node}(f)$
DPP Node	see Section 3	k_n hidden nodes	$\epsilon_{k_n}^{DPP\ Node}(f)$	$\hat{\epsilon}_{k_n}^{DPP\ Node}(f)$
Random Edge	Keep an edge w.p. c for each hidden node	k_e incoming edges per hidden node	$\epsilon_{k_e}^{Rand\ Edge}(f)$	$\hat{\epsilon}_{k_e}^{Rand\ Edge}(f)$

4.1 Comparing Node Pruning Methods

We theoretically show that the increment in GE due to DIVNET is less than that for random and importance node pruning methods, justifying the empirical findings of [29]. The proof proceeds with the following steps: (1) Theorem 1 provides a closed form expression of the GE after DPP node pruning. (2) Theorem 2 shows that: (a) GE of random node pruning is greater than GE of DPP node pruning (b) GE of random node pruning with reweighting is greater than GE of DIVNET (c) GE of importance node pruning is greater than GE of DIVNET.

Theorem 1. *Assume $(A1)-(A7)$. Let $k_n \leq M$ nodes are selected by the DPP Node pruning method,*

$$\epsilon_{k_n}^{DPPNode}(f) = (v^*)^2 \left[\frac{k_n}{6}\left(1-\frac{1}{Z}\right)^2 + \frac{M-k_n}{6}\right] \text{ and } \hat{\epsilon}_{k_n}^{DPPNode}(f) = (M-k_n) \times \frac{(v^*)^2}{6}. \tag{5}$$

Proof Idea of Theorem 1: Proof of the above theorem (details in the appendix C) is based on two factors: (1) Results from [10] assure that analyzing the *order parameters* is enough to obtain closed form of GE. (2) We exploit the observation that the expected kernel of the DPP node pruning is same as the order parameter Q (see appendix B for proof and Figure 2 E) which, following [10], is a block diagonal matrix with M blocks. By property of DPP, the pruning method will retain a subset of student hidden nodes with at most 1 hidden node from each block when $k_n \leq M$ (see Figure 2 H).

Remark 1. *As the expected DPP kernel is block-diagonal matrix, the stochasticity in subset selection via DPP does not impact GE when subset size is fixed and it only depends on size of pruned subsets.*

Remark 2. *Our theorem uses $k_n \leq M$, however, in practice the kernel may have non-zero off-diagonal entries when the assumption (A1) about input data is violated. As a result the probability of sampling a subset of size $k_n > M$ may be nonzero.*

Theorem 2. *Assume $(A1)-(A7)$. Then for $k_n \leq M$ we have,*

$$\mathbb{E}_f\left[\epsilon_{k_n}^{Rand\ Node}(f)\right] \geq \epsilon_{k_n}^{DPP\ Node}(f) \text{ and } \mathbb{E}_f\left[\hat{\epsilon}_{k_n}^{Rand\ Node}(f)\right] \geq \hat{\epsilon}_{k_n}^{DPP\ Node}(f) \tag{6}$$

and,

$$\epsilon_{k_n}^{Imp\ Node}(f) \geq \hat{\epsilon}_{k_n}^{DPP\ Node}(f). \tag{7}$$

Here the expectation is taken over the the subsets of hidden nodes of size k_n chosen u.a.r.

Remark 3. *Reweighting for DPP/random node pruning follow procedure in Section 2.2 of [29].*

Proof Idea of Theorem 2: In random and importance node pruning, two student nodes which learn the same teacher node may both survive after pruning with non-zero probability, unlike DPP node pruning (Figure 1 (B)). Hence, more teacher nodes may remain unexplained by the student network after random or importance node pruning, resulting in increased GE (details in appendix C).
Together, Theorem 1 and 2 gives theoretical guarantees for all empirical results of [29]. Importance node pruning with reweighting may be better than DIVNET and was not explored in [29].

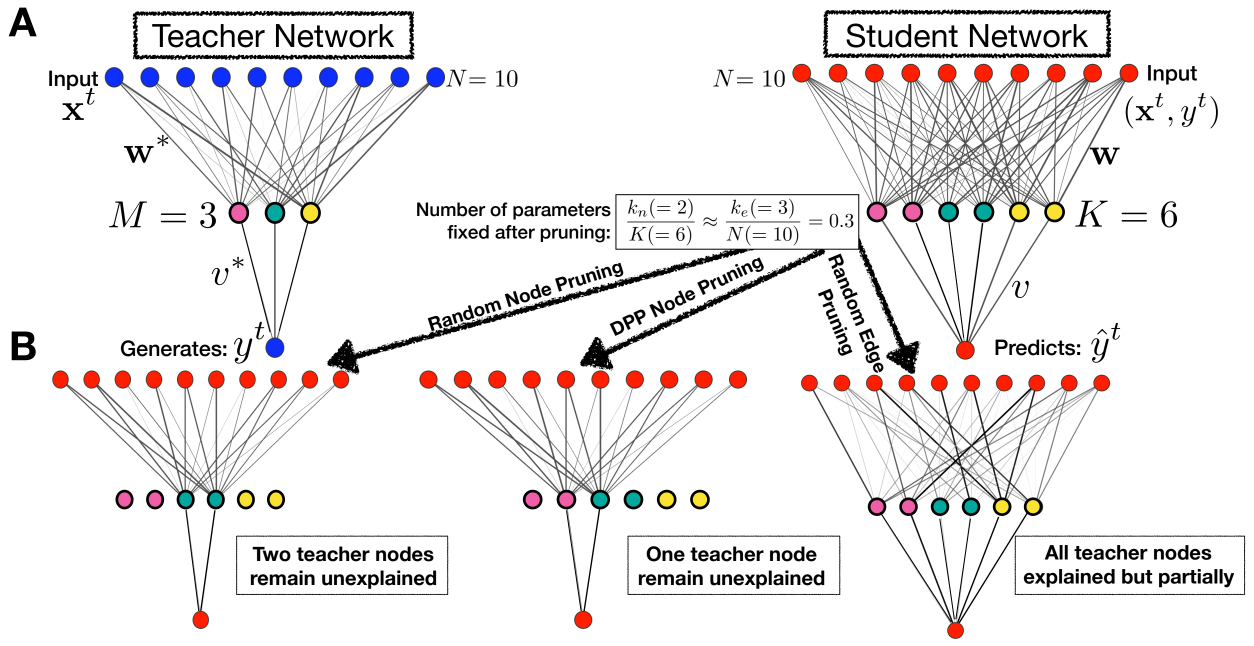
A
Teacher Network
Input $\mathbf{x}^t$
$N = 10$
$\mathbf{w}^*$
$M = 3$
v^*
Generates: y^t
Student Network
$N = 10$
Input $(\mathbf{x}^t, y^t)$
$\mathbf{w}$
$K = 6$
v
Predicts: $\hat{y}^t$
Number of parameters fixed after pruning: $\frac{k_n(=2)}{K(=6)} \approx \frac{k_e(=3)}{N(=10)} = 0.3$
Random Node Pruning
DPP Node Pruning
Random Edge Pruning
B
Two teacher nodes remain unexplained
One teacher node remain unexplained
All teacher nodes explained but partially

Figure 1: **(A)** Two layer teacher-student framework: A teacher neural network with 3 hidden nodes (left) and a student network with 6 hidden nodes (right). Input data (i.i.d) along with its label generated by teacher network are fed to student network to predict. **(B)** Intuitive example for 3 types of pruning on student network. For $k_n = 2$, random node pruning might only be able to explain 1 teacher hidden node, whereas DPP node pruning will always retain (partial) information about 2 teacher hidden nodes, hence preforms better. Random edge pruning retains sparse information about all 3 teacher nodes which is enough to outperform DPP node pruning. All notations follow Table 1.

4.2 Comparing Node and Edge Pruning Methods

In random edge pruning method, for each student hidden node, an incoming edge is kept with probability $c = \lim_{N\to\infty} \frac{k_e}{N}$. Majority of empirical studies throughout literature use random edge or node pruning as a baseline for empirical comparison (see papers in [4]) making it an obvious candidate for our theoretical comparisons as well. It has been shown empirically by [29] and theoretically by us that DPP node pruning is an above baseline node pruning method. In this section we show that baseline random edge pruning outperforms DPP node pruning which is consistent with the empirical observations that sparse models outperform dense models (section 3.2 of [4]). Specifically, here we show that GE after random edge pruning is less than GE after DPP node pruning. Our proof proceeds as follows: (1) Theorem 3 gives a closed form expression for the GE after random edge pruning (2) Theorem 4 then shows that GE of random edge pruning is less than GE of DPP node pruning.

Theorem 3. *Assume $(A1) - (A7)$. Consider the random edge pruning method with parameter $\lim_{N\to\infty} \frac{k_e}{N} = c$ (here c is a constant between 0 and 1). Then the GE $\epsilon_c^{Rand\ Edge}(\mathbb{E}[f])$ is,*

$$\frac{M(v^*)^2}{\pi}\left[\frac{1}{Z}\arcsin\frac{c}{1+c} + \left(1-\frac{1}{Z}\right)\arcsin\frac{c^2}{1+c} + \frac{\pi}{6} - 2\arcsin\frac{c}{\sqrt{2(1+c)}}\right]. \tag{8}$$

Remark 4. *Theorem 3 gives closed form for "GE of the expected network" after pruning instead of the "expected GE of the network" after pruning. However, in the thermodynamic limit ($N \to \infty$), the order parameters as in Section 3 are highly concentrated near their expected values and the two quantities hence become equal.*

Theorem 4. *Assume $(A1) - (A7)$. Let k_n and c satisfy (4), and $0 \leq c \leq \frac{1}{Z}$ and $Z \geq 4$. Then*

$$\epsilon_{k_n}^{DPP\ Node}(f) \geq \epsilon_c^{Rand\ Edge}(\mathbb{E}[f]). \tag{9}$$

Proof Idea of Theorem 4: When $k_n \leq M$, node pruned student network leaves at least $(M - k_n)$ teacher nodes unexplained, whereas after random edge pruning, student network can retain at least partial information about every teacher node (see Figure 1 (B)). After a pruning routine, the sum of partial information about all teacher nodes in an edge pruned student network dominates the sum of information for the explained subset of teacher nodes in a node pruned student network.

Observations: From Theorem 2 and 4, we conclude that random edge pruning outperforms random node pruning. Further, using Theorem 2 and the intuition that importance edge pruning is better than random edge pruning, we expect that importance edge pruning will outperform importance node pruning. Figure 2 confirms this empirically in the teacher student setup. These observations leads to the conjecture that for a fixed pruning method, edge pruning outperforms node pruning.

Conjecture 1. *Assume* $(A1) - (A7)$*. Let* k_n *and* c *satisfy* (4) *and Prune denotes a fixed pruning method (e.g. Rand, Imp, DPP etc.) which can be applied to both node and edge. Then,* $\exists c_\epsilon \in (0, 1]$ *such that for* $0 \leq c \leq c_\epsilon$,

$$\epsilon_{k_n}^{Prune\ Node}(f) \geq \epsilon_c^{Prune\ Edge}(f). \tag{10}$$

Together, Theorem 3, 4 and Conjecture 1 are consistent with empirical observations of [4]: sparse networks after edge pruning perform better on the unseen test data than dense networks after node pruning with fixed number of parameters. To the best of our knowledge, [4] based their claims from empirical observations of pruning studies in which the pruned networks were not reweighted. This motivated our choice of comparing GE for DPP node pruning and random edge pruning without any reweighting. However, with reweighting from [29], GE of DPP node pruning will be less than GE of random edge pruning, highlighting the impact of reweighting proposed by [29] (proof and details in appendix C). We find that GE analysis on teacher-student setup is not only flexible for various pruning methods but also produces results that are consistent with empirical observations on real data (Section 6.2). This framework can be extended to theoretically understand other pruning methods which are outside the scope of this work. Yet, in its current form, it can only be used to compare pruning methods where networks are not retrained after pruning.

5 Edge Pruning using Determinantal Point Process

We theoretically showed (1) random edge pruning is better than DPP node pruning (2) DPP node pruning beats random and importance node pruning. Together this motivated the idea of *DPP edge pruning* which models diversities on synaptic levels instead of neuronal levels.

5.1 DPP Kernel for Edge Pruning

We follow [29] to compute the amount of information captured by an edge about the training data. Recall from Section 3 that for node v_j^l of layer l, the activation is given as $\boldsymbol{a}_j^l = g\left(\sum_{i=1}^{n_{l-1}} w_{ij}^{l-1} \boldsymbol{a}_i^{l-1}\right)$, where w_{ij}^{l-1} is the weight of the edge from i^{th} node of layer $l-1$ to j^{th} node of layer l. Hence the information carried from layer $l-1$ to node v_j^l of layer l through the edge e_{ij}^{l-1} is $w_{ij}^{l-1}\boldsymbol{a}_i^{l-1}$. We choose a subset of incoming edges for node v_j^l such that it preserves diverse information about the input. The DPP kernel $\boldsymbol{L}^j$ for the set of incoming edges of v_j^l is defined as,

$$\boldsymbol{L}^j = \boldsymbol{L'}^{(j)} + \epsilon \boldsymbol{I}, \tag{11}$$

where $\boldsymbol{L'}^{(j)}_{st} = \exp(-\beta \left\| w_{sj}^{l-1}\boldsymbol{a}_s^{l-1} - w_{tj}^{l-1}\boldsymbol{a}_t^{l-1} \right\|^2)$. Here β is a hyperparameter and ϵ denotes small perturbations to the diagonal of the kernel matrix making it strictly positive definite (as in [29]).

5.2 Retrieving Pruned Information by Reweighting

We lose information passed from one layer to the next by removing a subset of edges between them after pruning. To maximize the total amount of information preserved, we reweight the surviving edges. We denote $S_j = \{i_1, \ldots, i_{k_e}\}$ as the set of incoming edges chosen for the node v_j^l using k_e-DPP. Denote $\hat{w}_{ij}^{l-1} = w_{ij}^{l-1} + \delta_{ij}^{l-1}$ to be the new weight of the edges after reweighting. Hence to minimize the lost information after pruning, we minimize:

$$\left\| \sum_{i=1}^{n_{l-1}} w_{ij}^{l-1}\boldsymbol{a}_i^{l-1} - \sum_{i \in S_j} \hat{w}_{ij}^{l-1}\boldsymbol{a}_i^{l-1} \right\|_2 = \left\| \sum_{i \in \bar{S}_j} w_{ij}^{l-1}\boldsymbol{a}_i^{l-1} - \sum_{i \in S_j} \delta_{ij}^{l-1}\boldsymbol{a}_i^{l-1} \right\|_2 .$$

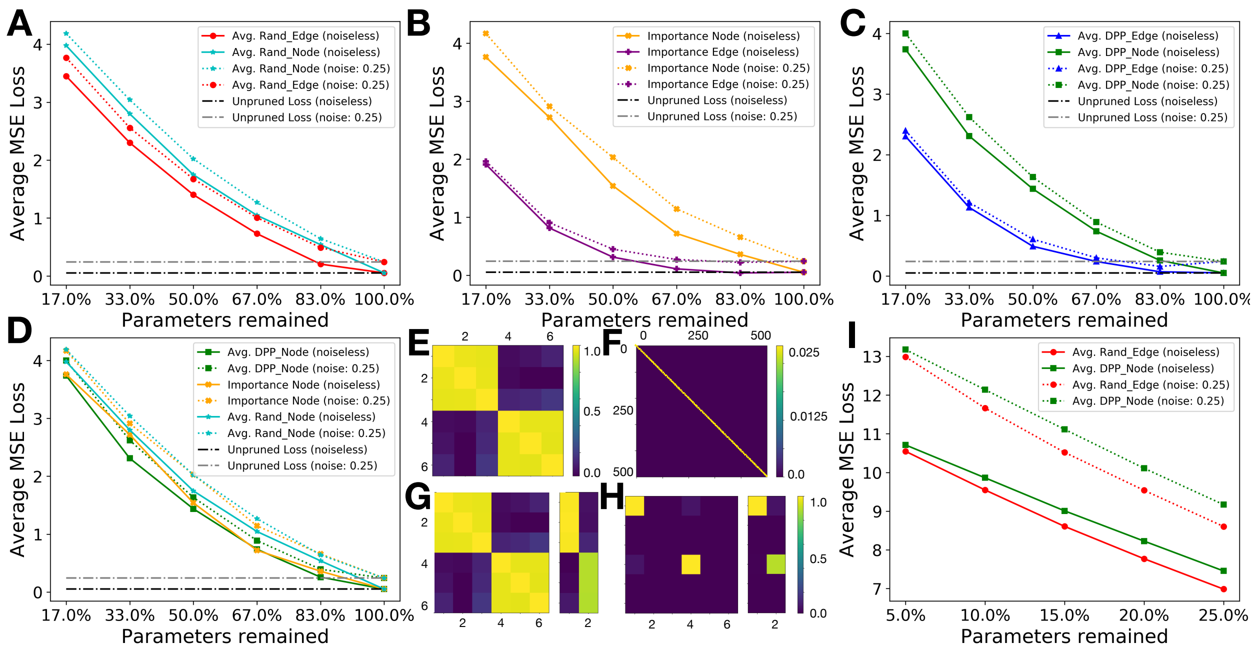

Figure 2: Simulation results in teacher student setup, $M = 2$ and $K = 6$ for (A-H). **(A-C)** Edge pruned networks perform better than node pruned networks in all 3 types of pruning methods (random (A), importance (B) and DPP (C)), validating Conjecture 1. **(D)** DPP Node pruning performs better than importance and random node pruning (Theorem 2) **(E)** The kernel of DPP node pruning is same as Q **(F)** The kernel of DPP edge pruning is a diagonal matrix. **(G)** Order parameters, Q (same as (E)) and R of the unpruned student network. **(H)** When only keeping 2 nodes, DPP node pruned student network keeps one from each block shown in (E). **(I)** Baseline random edge pruning beats DPP node pruning (Theorem 4). For (I), $M = 5$ and $K = 20$.

This can be rewritten as a linear regression problem i.e.,$\min_{\boldsymbol{x}} \left\|\boldsymbol{y} - A_{l-1} \cdot \boldsymbol{x}^j\right\|_2$, where s^{th} column of A_{l-1} is $\boldsymbol{a}_{i_s}^{l-1}$, $\boldsymbol{x}^j = [\delta_{i_1 j}, \cdots, \delta_{i_{k_e} j}]^T$, and $\boldsymbol{y} = \sum_{i \in \bar{S}_j} w_{ij}^{l-1} \boldsymbol{a}_i^{l-1}$. The closed form solution of this equation is $\boldsymbol{x} = \left(A_{l-1}^T A_{l-1}\right)^{-1} A_{l-1}^T \boldsymbol{y}$.
In node pruning, if a node is deleted from layer l, edges from $l-1$ to l and l to $l+1$ are deleted, affecting both incoming and outgoing information from layer l. Therefore, the focus is to minimize loss in outgoing information from layer l to $l+1$ as is ensured by reweighting of [29]. In edge pruning, edges between layer $l-1$ to l get deleted which we try to compensate in our new reweighting scheme by minimizing incoming information loss.
While DPP node pruning requires only one kernel per layer [29], DPP edge pruning will have n_l DPP kernels for l^{th} layer. However, DPP kernels require one-time computation and are not recomputed to predict labels for unseen data. Our method falls under fast compression techniques that do not undergo retraining and outperforms competing methods in the same category on real data (Section 6.2). While DPP edge pruning is complementary to prior work, its novelty lies in its theory motivated definition. We consolidate the understanding of previous empirical findings and implications of observations in the brain from a theoretical perspective to propose a new pruning method. Note that while DPP edge pruning with reweighting inherits all benefits of DIVNET discussed in Section 3.3 and 4 of [29], it performs better than the latter on real data (Section 6.2). Few important properties inherited by DPP edge kernels due to similar definitions to DIVNET are: (1) The DPP kernel and the reweighting coefficients can also be learned from subsampled training data which causes a trade-off between performance, space complexity, and time complexity. (2) DPP edge pruning with reweighting can be incorporated into other methods of network compression to increase diversity.

6 Experiments

6.1 Simulations

We run the DPP edge/node, random edge/node, and importance edge/node pruning simulations under the teacher-student setup. For all the simulations, we sampled the 800000 i.i.d input samples from $\mathcal{N}(0,1)$ as training data and 80000 as testing data. Following notations from Table 1, we set $M = 2$, $K = 6$, $N = 500$, and $v^* = 4$. The first layer teacher network weights $\boldsymbol{w}^*$ and all the student network parameters $\theta = \{\boldsymbol{w}, \boldsymbol{v}\}$ were drawn independently from $\mathcal{N}(0,1)$ as initialization. We choose learning rate $\eta = 0.50$, and it is scaled to $\frac{\eta}{\sqrt{N}}$ for $\boldsymbol{w}$ and $\frac{\eta}{N}$ for $\boldsymbol{v}$. We run the simulations for both noiseless ($\sigma = 0$ in (1)) and noisy ($\sigma = 0.25$) output labels. For comparisons between node and edge pruning, we use the node-to-edge ratio $[1:83, 2:166, 3:250, 4:333, 5:417, 6:500]$ to keep the number of parameters the same, given $N = 500$, $K = 6$, and $M = 2$. In addition, we run the same simulation with $K = 5$ and $M = 20$, see Figure 2I. For other simulation details and results, see E. Note that no pruning method undergoes reweighting for reported simulation results which we therefore use to verify and validate our theoretical results without reweighting.
Key Observations: (1) The expected kernel of the DPP node pruning and the Q matrix are the same which we exploit for Theorem 1. (2) For $k_n = 2$ and $M = 2$, DPP node pruning chooses exactly one node from each of the diagonal block of the kernel (see Figure 2H) which validates Theorem 1. (3) DPP node pruning outperforms random and importance node in both noisy and noiseless case (see Figure 2D) which confirms Theorem 2. (4) Random edge pruning is better than DPP node pruning for $c \leq \frac{1}{Z}$ with $Z = 4$ and $M = 5$ in both noisy and noiseless cases (see Figure 2I), validating Theorem 4. (5) DPP edge pruning outperforms all the other pruning methods considered for both noisy and noiseless cases. (6) In DPP edge pruning method, a subset of incoming edges (of size k_e) is chosen using the DPP kernel. The expected kernel of DPP edge pruning method is a diagonal matrix with corresponding edge weights as the diagonal entries due assumption (A1) about input data (see appendix B for proof). This defines an independent distribution on the set of incoming edges of a hidden node (see Figure 2F). Therefore, the DPP distribution selects an incoming edge with probability proportional to its weights, hence, edges with higher weights have higher probability to survive in this pruning. (7) In DPP edge pruning, important and diverse edges with higher weights have higher probability to survive than under other edge pruning methods. As a result the GE is lower for DPP edge pruning and even goes below GE for the unpruned network (see Figure 2C). (8) We see Conjecture 1 holds for random, importance, and DPP edge and node pruning (see Figure 2A,B,C).

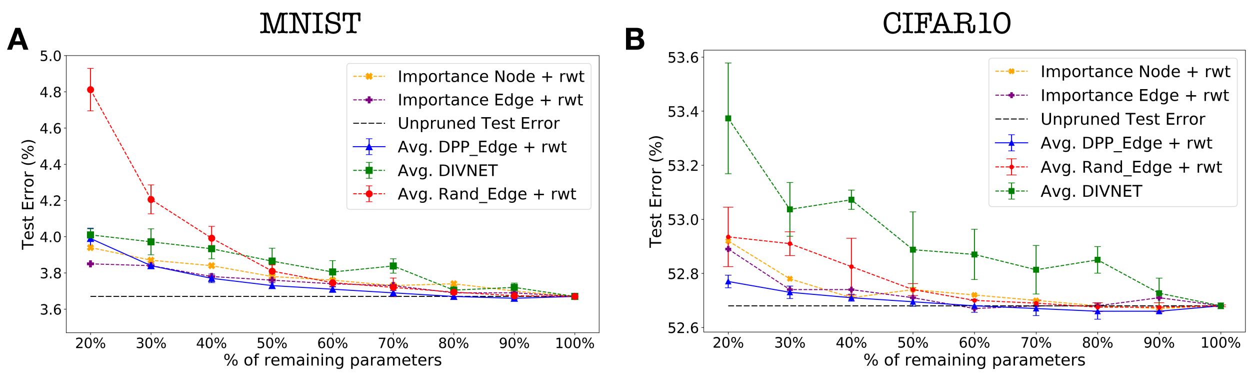

Figure 3: **(A-B)** Comparing DPP Edge pruning with other pruning methods on the `MNIST` (A) and `CIFAR10` (B) dataset. Horizontal axis represents the percentage of remaining parameters in 1^{st} layer after pruning. The vertical axis shows corresponding test error. DPP Edge pruning outperforms all other pruning methods considered. It also produces lower test error than the unpruned network for certain % (at 90% for `MNIST` and 70-90% for `CIFAR10`) without any retraining.

6.2 Experiment on Real Data

In this section, we compare our proposed DPP edge pruning with reweighting, DIVNET by [29], random edge pruning with reweighting, and importance edge/node pruning with reweighting on the `MNIST` [21] and `CIFAR10` [18] datasets. We used the exact same network architectures as in Table 1 of [29] for `MNIST` and `CIFAR10`, respectively. Following [29], we performed all pruning methods on the first layer. We compare the number of parameters as $k_e = \frac{k_n(d_{\text{input}}+h_2)}{h_1} - h_2$ where k_e is the number of edges kept for each node in edge pruning, and k_n is the number of nodes kept in the hidden layer for node pruning; d_{input}, h_1, and h_2 represent the dimension of the input, size of the first hidden layer, and size of the second hidden layer, respectively. As in [29], $h_1 = h_2$. We trained our model until the training error reaches predefined thresholds (Table 1 in [29]) and then perform the pruning. All edge pruning methods use our reweighting scheme in 5.2 while all node pruning methods use reweighting from [29]. For hyperparameters and other details, see F.
Key observations: We compare the test errors for the pruned networks with respective reweighting schemes for both datasets as in Figure 3. We observe that (1) DPP edge pruned network significantly outperforms all other pruning methods, including DIVNET (consistent with Conjecture 1). (2) Interestingly, we find that importance node pruning with reweighting performs better than DIVNET, which was not explored in [29]. (3) Furthermore, unlike other pruning methods, our sparse DPP edge pruned network generalizes better than the unpruned dense network for both the datasets (consitent with the *teacher student* framework, see Figure 2C). In literature, such sparse sub-networks of a dense network with better generalization ability is achieved via iterative pruning method [28], which is time consuming. However, our approach gives rise to such sparse network without reinitializing and retraining but only reweighting (therefore, loosely connecting to Lottery Ticket Hypothesis [8] but not exactly) adding to the novelty of our approach.

7 Conclusion

Our work is the first to develop theoretical understanding of empirical studies of pruning methods in feed forward neural networks. Based on our theoretical results, we proposed DPP edge pruning which outperforms comparable pruning methods on real data. Our work consolidates the understanding of diversity-based node and edge pruning both theoretically and empirically. When comparing two neural networks, using the number of parameters may not always be the optimal choice, instead, measuring the *capacity* and *expressiveness* of neural networks [2] can provide new insights. Finally, our theoretical and empirical observations can motivate informed and insightful use of pruning methods(especially DPP based), on large networks like CNNs and RNNs in future work.

References

[1] Madhu S Advani and Andrew M Saxe. High-dimensional dynamics of generalization error in neural networks. *arXiv preprint arXiv:1710.03667*, 2017.

[2] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 224–232. JMLR. org, 2017.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[4] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? *arXiv preprint arXiv:2003.03033*, 2020.

[5] Gal Chechik, Isaac Meilijson, and Eytan Ruppin. Synaptic pruning in development: a computational account. *Neural computation*, 10(7):1759–1777, 1998.

[6] Li Deng and Yang Liu. *Deep learning in natural language processing.* Springer, 2018.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[8] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[9] Elizabeth Gardner and Bernard Derrida. Three unfinished works on the optimal storage capacity of networks. *Journal of Physics A: Mathematical and General*, 22(12):1983, 1989.

[10] Sebastian Goldt, Madhu Advani, Andrew M Saxe, Florent Krzakala, and Lenka Zdeborová. Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. In *Advances in Neural Information Processing Systems*, pages 6979–6989, 2019.

[11] Scott Gray, Alec Radford, and Diederik P Kingma. Gpu kernels for block-sparse weights. *arXiv preprint arXiv:1711.09224*, 3, 2017.

[12] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[13] Babak Hassibi and David G Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in neural information processing systems*, pages 164–171, 1993.

[14] Tianxing He, Yuchen Fan, Yanmin Qian, Tian Tan, and Kai Yu. Reshaping deep neural network for fast decoding by node-pruning. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 245–249. IEEE, 2014.

[15] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017.

[16] Kumar Joag-Dev, Frank Proschan, et al. Negative association of random variables with applications. *The Annals of Statistics*, 11(1):286–295, 1983.

[17] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. *arXiv preprint arXiv:1802.08435*, 2018.

[18] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[20] Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

[21] Yann LeCun, Corinna Cortes, and Chris Burges. Mnist handwritten digit database. 2010. *URL http://yann. lecun. com/exdb/mnist*, 3(1), 2010.

[22] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

[23] Namhoon Lee, Thalaiyasingam Ajanthan, Stephen Gould, and Philip HS Torr. A signal propagation perspective for pruning neural networks at initialization. *arXiv preprint arXiv:1906.06307*, 2019.

[24] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

[25] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

[26] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234:11–26, 2017.

[27] Odile Macchi. The coincidence approach to stochastic point processes. *Advances in Applied Probability*, 7(1):83–122, 1975.

[28] Eran Malach, Gilad Yehudai, Shai Shalev-Shwartz, and Ohad Shamir. Proving the lottery ticket hypothesis: Pruning is all you need. *arXiv preprint arXiv:2002.00585*, 2020.

[29] Zelda Mariet and Suvrit Sra. Diversity networks: Neural network compression using determinantal point processes. *arXiv preprint arXiv:1511.05077*, 2015.

[30] Mark Mayford, Steven A Siegelbaum, and Eric R Kandel. Synapses and memory storage. *Cold Spring Harbor perspectives in biology*, 4(6):a005751, 2012.

[31] Nancy A O'rourke, Nicholas C Weiler, Kristina D Micheva, and Stephen J Smith. Deep molecular diversity of mammalian synapses: why it matters and how to measure it. *Nature Reviews Neuroscience*, 13(6):365–379, 2012.

[32] David Saad and Sara A Solla. Exact solution for on-line learning in multilayer neural networks. *Physical Review Letters*, 74(21):4337, 1995.

[33] David Saad and Sara A Solla. On-line learning in soft committee machines. *Physical Review E*, 52(4):4225, 1995.

[34] David Saad and Sara A Solla. Learning with noise and regularizers in multilayer neural networks. In *Advances in Neural Information Processing Systems*, pages 260–266, 1997.

[35] Tara N Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6655–6659. IEEE, 2013.

[36] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[37] Vladimir N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

A GE in Two Layer Network

For the theoretical analysis we consider the following assumptions from [10]
(A1) If $\boldsymbol{x} = (x_1, \dots, x_N)$ is an input then $x_i \in \mathcal{N}(0, 1)$. Also, $N \to \infty$.
(A2) Both the teacher and the student networks have only one hidden layer.
(A3) M, K denotes the number of hidden nodes for the teacher and student network respectively and $K \geq M$ and $K = Z \cdot M$ where $Z \in \mathbb{Z}^+$.
(A4) The activation in the hidden layer is sigmoidal for both teacher and student network.
(A5) The output $\in \mathbb{R}$ (i.e., regression problem).
(A6) The order parameters (see section 3) satisfy the ansatz as in (S58) - (S60) of [10].
(A7) No noise is added to the labels generated by the teacher network, i.e., $\sigma = 0$ in (1).
With the above assumptions, authors of [10] gave a closed form of the GE as follows:

$$\epsilon_g = f_1(Q) + f_2(T) - f_3(R, Q, T) \tag{12}$$

where,

$$f_1(Q) = \frac{1}{\pi} \sum_{i,k} v_i v_k \arcsin \frac{Q_{ik}}{\sqrt{1+Q_{ii}}\sqrt{1+Q_{kk}}} \tag{13}$$

$$f_2(T) = \frac{1}{\pi} \sum_{n,m} v^*_n v^*_m \arcsin \frac{T_{nm}}{\sqrt{1+T_{nn}}\sqrt{1+T_{mm}}} \tag{14}$$

$$f_3(R, Q, T) = \frac{2}{\pi} \sum_{i,n} v_i v^*_n \arcsin \frac{R_{in}}{\sqrt{1+Q_{ii}}\sqrt{1+T_{nn}}} \tag{15}$$

where Q, R, T are the order parameters as defined in section 3.

B Properties of DPP Kernel

In section 3 we see that each node in the hidden layer of a student network carries certain amount of information about the training data and it is captured in a vector form. We create an information matrix by accumulating the information vectors of these hidden nodes. For simplicity of theoretical analysis, we have considered the kernel as the inner product of the information matrix. In the thermodynamic limit, the inner product is divided by the input dimension. Formally, if $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ are the information at i^{th} hidden node and j^{th} hidden node respectively, then

$$L_{ij} = \frac{1}{N}\frac{1}{n}\boldsymbol{h}_i^T \boldsymbol{h}_j$$

where n is the total number of training examples. It can be seen that the analysis for the kernel defined in Section 3 and 5.1 are similar. Note that all analyses are for the student network trying learn from the teacher network. Refer to Table 1 for details of notations.

Lemma 1. *Assume (A1) - (A7). Then the expected kernel of DPP Node for the hidden layer is the order parameter Q.*

Proof of Lemma 1. For the two-layer teacher-student setup, the hidden layer gets information $(\boldsymbol{h}_1, \dots, \boldsymbol{h}_K)$ from the input layer, where $\boldsymbol{h}_i = (h_{i1}, \dots, h_{in})$ and $h_{ij} (= t_j^T \boldsymbol{w}_i)$ is the information at i^{th} hidden node on j^{th} input data (t_j). Hence,

$$\boldsymbol{h}_i^T \boldsymbol{h}_j = \sum_{k=1}^{n} h_{ik} h_{jk} = \sum_{k=1}^{n} t_k^T \boldsymbol{w}_i \cdot t_k^T \boldsymbol{w}_j = \sum_{k=1}^{n} \boldsymbol{w}_i^T t_k \cdot t_k^T \boldsymbol{w}_j = \sum_{k=1}^{n} \boldsymbol{w}_i^T (t_k t_k^T) \boldsymbol{w}_j$$

But for the given input distribution (i.i.d. Gaussian), $\mathbb{E}[t_k t_k^T] = \boldsymbol{I}_{N \times N}$. Hence, $\lim\limits_{N\to\infty} \mathbb{E}[L_{ij}] = \lim\limits_{N\to\infty} \mathbb{E}[\frac{1}{N}\frac{1}{n}\boldsymbol{h}_i^T \boldsymbol{h}_j] = \lim\limits_{N\to\infty} \frac{1}{N}\boldsymbol{w}_i^T \boldsymbol{w}_j = Q_{ij}$, and we have the lemma. □

From [10] we know that Q is a block diagonal matrix where each "block" (or "group" used interchangeably henceforth) refers to the set of student hidden nodes that represent (explian/learn) one particular teacher hidden node.

Lemma 2. *Assume (A1) - (A7). Then the expected kernel of DPP Edge for the s^{th} hidden node is*

$$\mathbb{E}[L^s_{ij}] = \begin{cases} \frac{1}{N} w_{is}^2 & \textit{if } i = j \\ 0 & \textit{otherwise} \end{cases}$$

Proof of Lemma 2. For the s^{th} hidden node, we choose a subset of incoming edges w.r.t. a DPP. The representation for the j^{th} incoming edge is $\boldsymbol{u}^s_j = (u^s_{1j}, \ldots, u^s_{nj})$, where $u^s_{kj} = t_{kj} w_{js}$. Hence

$$L^s_{ij} = \frac{1}{N}\frac{1}{n}\sum_{k=1}^{n} u^s_{ki} u^s_{kj} = \frac{1}{N}\frac{1}{n}\sum_{k=1}^{n} t_{ki} w_{is} t_{kj} w_{js}$$

As the input is drawn from i.i.d. Gaussian distribution, it can be seen that $\mathbb{E}[t_{ki}t_{kj}] = 1 \iff i = j$. Hence we have

$$\mathbb{E}[L^s_{ij}] = \begin{cases} \frac{1}{N} w_{is}^2 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

And we have the lemma. □

This lemma basically states that the DPP edge method under the teacher student setup has a diagonal kernel.

C Proof of the Theorems

Proof of Theorem 1 and 2. Let $H_R = \{h_{i_1}, \ldots, h_{i_{k_n}}\}$ be the set of selected nodes by DPP Node pruning method. Recall from [10] that every student hidden node specializes in learning a teacher node. Denote $t(h)$ to be the teacher node learnt by h. $S_m \subseteq H_R$ be the set of selected hidden nodes of the pruned network which learnt the m^{th} teacher node , i.e., $S_m = \{h \in H_R | t(h) = t_m\}$ (t_m is the m^{th} teacher node). Hence, $prn = |\{\mathbb{1}(|S_m| > 0) | 1 \leq m \leq M\}|$ is the number of teacher nodes explained by the pruned network and W.L.O.G. we can assume that $t_1, \ldots, t_{prn}$ are those set of teacher nodes. Let $l_1, \ldots, l_{prn}$ be the number of student nodes in the pruned network which learn the corresponding teacher node. Note that, $\sum_{i=1}^{prn} l_i = k_n$ and $l_i \leq Z$ (where Z is the number of student nodes dedicated to learn a single teacher node in the unpruned network) for all i. Applying Lemma 3 directly we can see that the GE for the pruned network is

$$\frac{(v^*)^2}{6}\left[\sum_{i=1}^{prn}\left(1 - \frac{l_i}{Z}\right)^2\right] + \frac{(M - prn)(v^*)^2}{6} \tag{16}$$

The first part of (16) is the GE for the group whose corresponding teacher node is partially explained and the second part accounts for the GE due to unexplained teacher nodes (number of such teacher nodes are $M - prn$). From Lemma 1 we know that the expected kernel matrix for DPP Node pruning is the order parameter Q and it becomes a block diagonal matrix after the training converges, where size of each block is Z (which is also the number of student nodes dedicated to learn a single teacher node in the unpruned network). Because of the block diagonal property of the DPP kernel matrix, at most 1 student hidden node will be chosen from each block, i.e., $l_i = 1\ \forall i$. Hence, $prn = k_n$. From Lemma 3 we can see that the GE of node pruned network only depends on the number of student node survived in each block after pruning, and, for DPP node pruning, it is always 1 (given $k_n \leq M$). This is why there is no expectation in the GE term. So for DPP node pruning the GE is,

$$\epsilon_{k_n}^{DPP\ Node}(f) = (v^*)^2\left[\frac{k_n}{6}\left(1 - \frac{1}{Z}\right)^2 + \frac{M - k_n}{6}\right].$$

Each of the k_n student nodes in the pruned network learns a different teacher node. Consider one such teacher node and call it t_i. In the unpruned network, there are Z student hidden nodes which learn a single teacher node t_i, only one of which survives after DPP node pruning. The first part of the error is due to the removal of student nodes ($Z-1$ student nodes for each t_i). However, these errors can be retrieved by reweighting the survived student node. On the contrary, there are $M-k_n$ teacher nodes which don't have any representative (some student hidden node from the set of student nodes which specialized in this particular teacher node) in the pruned network. And the error (second part of the GE) due to those nodes can not be retrieved even after reweighting. Hence, the GE after reweighting becomes,

$$(M-k_n)\times\frac{(v^*)^2}{6}$$

Thus, we have the Theorem 1.

Next, we will prove Theorem 2. We will show, for any network pruned by Random Node, the GE is more than the expected GE of DPP Node pruning. Recall the randomly pruned network f discussed in the beginning of the proof. From Lemma 3 we can see that for node pruning the GE only depends on the number of nodes survived in each block. From (16) we have,

$$\begin{aligned}
&\epsilon_{k_n}^{Rand\ Node}(f)\\
&=\frac{(v^*)^2}{6}\left[\sum_{i=1}^{prn}\left(1-\frac{l_i}{Z}\right)^2\right]+\frac{(M-prn)(v^*)^2}{6}\\
&=\frac{(M-k_n)(v^*)^2}{6}+\sum_{i=1}^{prn}\left[(l_i-1)\frac{(v^*)^2}{6}+\frac{(v^*)^2}{6}\left(1-\frac{l_i}{Z}\right)^2\right]\\
&\geq\frac{(M-k_n)(v^*)^2}{6}+\sum_{i=1}^{prn}l_i\frac{(v^*)^2}{6}\left(1-\frac{1}{Z}\right)^2 \qquad (17)\\
&=\frac{(M-k_n)(v^*)^2}{6}+l\frac{(v^*)^2}{6}\left(1-\frac{1}{Z}\right)^2\\
&=\epsilon_{k_n}^{DPP\ Node}(f)
\end{aligned}$$

where (17) follows from the inequality below:

$$(l_i-1)\frac{(v^*)^2}{6}+\frac{(v^*)^2}{6}\left(1-\frac{l_i}{Z}\right)^2=l_i\frac{(v^*)^2}{6}\left[1+\frac{1}{Z^2}-\frac{2}{Z}\right]\geq l_i\frac{(v^*)^2}{6}\left(1-\frac{1}{Z}\right)^2$$

which proves the first part of Theorem 2. The proof for the reweighted network is similar.

In case of importance node pruning, the nodes with lowest absolute value of outgoing edges are dropped. Following [10] the outgoing weights of all the hidden teacher nodes are equal (we call it v^*). Also, from Lemma 4 we see that the sum of the weights of the outgoing edges of the student nodes which learn the same teacher node add up to the outgoing edge weight of the corresponding teacher hidden node. Moreover, we assume the ansatz $v_i = v_j$ when $i, j \in G_n$, where G_n denotes the set of student nodes which learn the same teacher node t_n. Hence, we can see that all the outgoing edges are approximately similar. We also verify this fact experimentally. Therefore, this defines an approximately uniform distribution on the set of hidden nodes. Hence, this is almost same as random node pruning and so the result follows from Theorem 2. □

Remark 5. *The comparison between performance of importance node pruning and DIVNET depends on the fact that all the outgoing edges of the teacher hidden nodes are equal. However, when the outgoing weights are not equal the importance pruning first selects student hidden nodes from a group whose corresponding teacher node has the highest weight. Once all the student nodes are selected from that group then it selects the group whose corresponding teacher node has second highest outgoing edge weight and the process continues. Because of this approach, even without reweighting a complete information about the teacher node is preserved in the pruned network. However, in DPP node pruning one candidate from each group (representing a particular teacher node) is selected first. But if a member is selected from a group then the reweighting method can recover the complete lost information for the corresponding group. Hence, DIVNET is able to preserve*

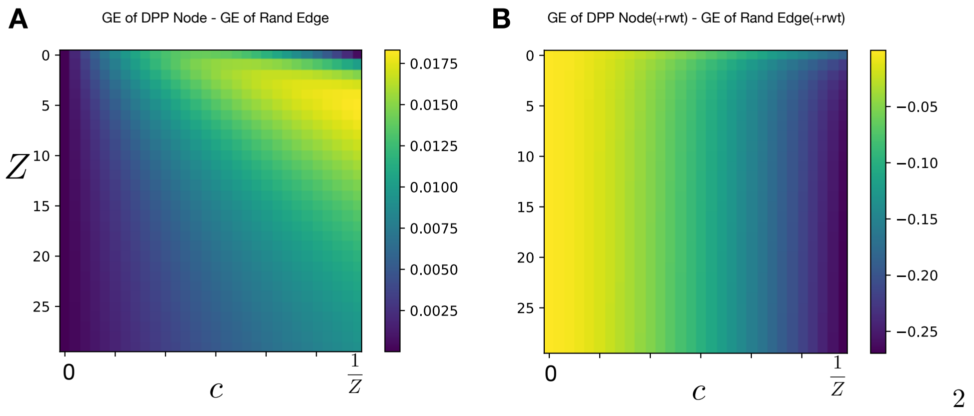

Figure 4: **(A)** Difference between the GE of DPP node pruning and Random edge pruning for $4 \geq Z \geq 30$. The matrix consist of only nonzero entries which proves that random edge pruning performs better than DPP node pruning when parameter count is same. **(B)** Difference between the GE of DPP node pruning with reweighting and Random edge pruning with reweighting for $4 \geq Z \geq 30$. The matrix consist of only negative entries which proves that random edge pruning can never perform better than DPP node pruning when reweighting is applied in the second layer.

information about more number of teacher hidden nodes than importance pruning which results in better performance.

Proof of Theorem 3. In this theorem, we will give the GE of the expected network pruned by the Random Edge method. Pruning is performed on the edges between input layer and the hidden layer. Hence, the order parameter changes. From Lemma 5, we have the order parameters of the expected network (call these Q', R', T'). However, the weights of the second layer remain unchanged. Putting these values in (13), (14) and (15) we have,

$$\begin{aligned} f_1(Q') &= \frac{1}{\pi}\sum_{i,k} v_i v_k \arcsin\frac{Q'_{ik}}{\sqrt{1+Q'_{ii}}\sqrt{1+Q'_{kk}}} \\ &= \frac{M(v^*)^2}{\pi}\arcsin\frac{c^2}{1+c} + \frac{M(v^*)^2}{Z\pi}\left[\arcsin\frac{c}{1+c} - \arcsin\frac{c^2}{1+c}\right] \end{aligned} \tag{18}$$

and,

$$\begin{aligned} f_3(R', Q', T') &= \frac{2}{\pi}\sum_{i,n} v_i v_n^* \arcsin\frac{R'_{in}}{\sqrt{1+Q'_{ii}}\sqrt{1+T'_{nn}}} \\ &= \frac{2M(v^*)^2}{\pi}\arcsin\frac{c}{\sqrt{2(1+c)}} \end{aligned} \tag{19}$$

Therefore, the GE of the expected network after Random Edge pruning is,

$$\frac{M(v^*)^2}{\pi}\left[\arcsin\frac{c^2}{1+c} + \frac{\pi}{6} - 2\arcsin\frac{c}{\sqrt{2(1+c)}}\right] + \frac{M(v^*)^2}{Z\pi}\left[\arcsin\frac{c}{1+c} - \arcsin\frac{c^2}{1+c}\right]$$

This proves the first part of the theorem. □

Proof of Theorem 4. Theorem 1 and 3 provide the closed form of the GE after DPP node pruning and random node pruning respectively. Using this closed form we plot $\epsilon_{k_n}^{DPP\ Node}(f) - \epsilon_c^{Rand\ edge}(f)$ in Figure 4 A. Here k_n and c satisfy (4), i.e., parameter count is same after two kinds of pruning. We can see for $Z \geq 4$ this value is ≥ 0 given $0 \leq c \leq 1.0/Z$, which proves the theorem. □

Remark 6. *Our results hold for $Z \geq 4$, where Z is the number of student nodes which learn the same teacher node. This is because in DPP node pruning at most 1 student node survives per group. As a result for larger Z the lost information per group is higher (in the scale of $\left(1 - \frac{1}{Z}\right)^2$).*

Next we state the impossibility result as discussed in Section 4. We will show that, no reweighting scheme in the second layer for random edge pruning which is based on scaling can beat DPP node pruning after reweighting. Formally we have the following:

Theorem 5. *Assume $(A1) - (A7)$. Let k_n and c satisfy (4), and $0 \leq c \leq \frac{1}{Z}$ and $Z \geq 4$. Assume the reweighting scheme for random edge in second layer such that, $\hat{v}_i = Av_i$. Then $\forall A \in \mathbb{R}$ we have,*

$$\hat{\epsilon}_{k_n}^{DPP\ Node}(f) \leq \hat{\epsilon}_c^{Rand\ Edge}(\mathbb{E}[f]) \tag{20}$$

Proof of Theorem 5. From Theorem 1 we know that the GE after rewighting the DPP node pruned network is

$$\frac{(v^*)^2}{6}(M - k_n) = \frac{M(v^*)^2}{6}(1 - Zc) \tag{21}$$

where c satisfies (4). Now for the given reweighting scheme in the hypothesis the GE for random edge pruning will be,

$$\frac{M(v^*)^2}{\pi}\left[A^2\left(\frac{1}{Z}\arcsin\frac{c}{1+c} + \left(1 - \frac{1}{Z}\right)\arcsin\frac{c^2}{1+c}\right) + \frac{\pi}{6} - 2A\arcsin\frac{c}{\sqrt{2(1+c)}}\right] \tag{22}$$

(22) can be viewed as a quadratic equation of A whose minimum correspond to the best reweighting scheme in the scaling family. In Figure 4 B we compare this minimum with (21). Formally we plotted $\hat{\epsilon}_{k_n}^{DPP\ Node}(f) - \hat{\epsilon}_c^{Rand\ Edge}(\mathbb{E}[f])$. It can be seen that this value is $-ve$ for all $0 \leq c \leq \frac{1}{Z}$, which implies GE of reweighted DPP node pruned network is always lower than reweighted random edge pruned network. □

D Proof of Lemmas

Lemma 3. *Assume (A1)-(A7). Let $t_1, \ldots, t_M$ denote the teacher hidden nodes and $l_1, \ldots, l_M$ denote the number of student hidden nodes in a node pruned network which learnt the corresponding teacher node. If $\sum_{m=1}^{M} l_m \leq M$, then the GE of this node pruned network is,*

$$\frac{(v^*)^2}{6}\left[\sum_{m=1}^{M}\left(1 - \frac{l_m}{Z}\right)^2\right].$$

Proof. Let $G_1, \ldots, G_M$ be the subsets of student nodes such that all student nodes in G_m learn the m^{th} teacher node. From the assumption we have, $|G_m| = Z$ for all m. After pruning, a subset $P_m \subseteq G_m$ is chosen, and $|P_m| = l_m$. Denote the order parameters of the pruned network as Q', R', T'. For node pruning we can see that

$$Q'_{ik} = \begin{cases} Q_{ik} & \text{if } \exists m \text{ s.t. } h_i \in P_m \text{ and } h_k \in P_m \\ 0 & \text{otherwise} \end{cases}$$

Also, for the unpruned network we have

$$Q_{ik} = \begin{cases} 1 & \text{if } \exists m \text{ s.t. } h_i \in G_m \text{ and } h_k \in G_m \\ 0 & \text{otherwise} \end{cases}$$

Now from (12) we can break down the GE into three parts. From (13), (14) and (15) we have,.

$$
\begin{aligned}
f_1(Q') &= \frac{1}{\pi}\sum_{i,k} v_i v_k \arcsin\frac{Q'_{ik}}{\sqrt{1+Q'_{ii}}\sqrt{1+Q'_{kk}}}, \\
&= \frac{1}{\pi}\sum_{n=1}^{M}\sum_{i,k\in P_n} v_i v_k \arcsin\frac{1}{2}, \qquad (23)\\
&= \frac{1}{\pi}\sum_{n=1}^{M}\sum_{i,k\in P_n} v_i v_k \frac{\pi}{6}, \\
&= \frac{1}{6}\sum_{n=1}^{M}\left(\sum_{i\in P_n} v_i\right)^2, \\
&= \frac{(v^*)^2}{6}\sum_{n=1}^{M}\left(\frac{l_i}{Z}\right)^2 \qquad (24)
\end{aligned}
$$

(23) follows from the fact that h_i and h_k belong to the same group G_n. So we have,

$$
\frac{Q'_{ik}}{\sqrt{1+Q'_{ii}}\sqrt{1+Q'_{kk}}} = \frac{1}{\sqrt{2}\sqrt{2}} = \frac{1}{2}
$$

We can also see that (24) follows from Lemma 4 and the ansatz $v_i = v_j$ when $i, j \in G_n$. The order parameters T_{nm} doesn't change after pruning, and so we have,

$$
\begin{aligned}
f_2(T') &= \frac{1}{\pi}\sum_{n,m} v^*_n v^*_m \arcsin\frac{T_{nm}}{\sqrt{1+T_{nn}}\sqrt{1+T_{mm}}}, \\
&= \frac{1}{6}\sum_{n=1}^{M}(v^*)^2 \qquad (25)
\end{aligned}
$$

And similarly,

$$
\begin{aligned}
f_3(R',Q',T') &= \frac{2}{\pi}\sum_{i,n} v_i v^*_n \arcsin\frac{R'_{in}}{\sqrt{1+Q'_{ii}}\sqrt{1+T'_{nn}}}, \\
&= \frac{2}{\pi}\sum_{n=1}^{M} v^*_n \sum_{i\in P_n} v_i \arcsin\frac{1}{2}, \\
&= \frac{2}{6}\sum_{n=1}^{M} v^*_n \sum_{i\in P_n} v_i. \qquad (26)
\end{aligned}
$$

Then from (24),(25) and (26) the GE of node pruning is,

$$
\frac{(v^*)^2}{6}\left[\sum_{m=1}^{M}\left(1-\frac{l_m}{Z}\right)^2\right]. \qquad (27)
$$

Hence we have the lemma. □

Intuitively, this lemma states that for teacher hidden node t_n if l_n student hidden nodes survive after node pruning, then the fraction of information lost due to the deletion of nodes is $1 - \frac{l_n}{Z}$, where Z is the number of student nodes learn a particular teacher node in the unpruned network.

Lemma 4. *Let v^* denotes the weight of the second layer of the teacher network and $\{v_1, \cdots, v_K\}$ be the weights of the student network after convergence. Then in the noiseless case for all n we have,*

$$
v^* = \sum_{i\in G_n} v_i
$$

Proof of Lemma 4. From $(S36)$ of [10] we have,

$$\begin{aligned}\frac{dv_i}{dt} &= \eta_v \left[\sum_{n=1}^{M} v_n^* I_2(i,n) - \sum_{j=1}^{K} v_j I_2(i,j)\right] \\ &= \eta_v \arcsin\frac{1}{2}\left[v^* - \sum_{j\in G_n} v_j\right]\end{aligned}$$

Hence, a fixed point (in terms of v_i's) of the ODE is,

$$\{(v_1,\ldots,v_K)| \sum_{i\in G_n} v_i = v^*, \forall 1 \le n \le M)\}$$

□

Intuitively, this lemma states that the sum of the outgoing edges of the student hidden nodes which learn a particular teacher hidden node is approximately equal to the weight of the outgoing edge of that teacher hidden node.

Lemma 5. *Let Q, R, T are the order parameters of the unpruned network, and Q', R', T' are the respective order parameters after applying the Random Edge pruning where c fraction of the edges are kept. Then we have the following:*

- $$\mathbb{E}[Q'_{ik}] = \begin{cases} cQ_{ik} & \textit{if } i = k \\ c^2 Q_{ik} & \textit{otherwise} \end{cases}$$
- $\mathbb{E}[R'_{st}] = cR_{st}$
- $T'_{mn} = T_{mn}$

Proof. In case of Random Edge pruning each edge is kept with probability c. Then we have,

$$\mathbb{E}[Q'_{st}] = \frac{1}{N}\sum_{i=1}^{N} c\cdot w_{is} \times c\cdot w_{it} = c^2\frac{1}{N}\sum_{i=1}^{N} w_{is}w_{it} = c^2 Q_{st}$$

and

$$\mathbb{E}[Q'_{ss}] = \frac{1}{N}\sum_{i=1}^{N} c^2\cdot w_{ss}^2 = c^2 Q_{ss}.$$

Similarly,

$$\mathbb{E}[R'_{st}] = \frac{1}{N}\sum_{i=1}^{N} c\cdot w_{is}w_{it}^* = cR_{st}$$

The teacher node is not affected by the pruning. So T is not modified by the pruning process. This proves the lemma. □

Intuitively, this lemma states that the order parameters of the pruned network using random edge pruning is a scaled version of the order parameters of the unpruned networks. However, the scaling of diagonal elements are different from that of off-diagonal elements (for more see Figure 6 A).

E Simulation Details

In total, 10 rounds of simulations are run for each of the 6 pruning methods in Section 6.1, and we report the average and standard deviations (as error bars). The standard deviations are negligible (in the magnitude of 10^{-3}). A *round* is the entire process of generating a new teacher network with datasets, training the student from scratch, performing pruning and finally testing with the pruned network. For DPP and random methods, we sampled 100 masks per round and reported the average performance in each round. Given $M = 2$ and $K = 6$, we tried pruning with $[1, 2, 3, 4, 5]$ nodes (and the equivalent number of edges) left in the student, respectively. We keep the total number of weights same to compare different pruning methods. The node-to-edge ratio, given $N = 500$, $K = 6$, and $M = 2$, is $[1 : 83, 2 : 166, 3 : 250, 4 : 333, 5 : 417]$. This is calculated, for the teacher-student setup (single output node) specifically, as $k_e = \frac{k_n(1+N)-K}{K}$. We grid-searched η in the range of $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7]$ and found 0.50 to be the optimal. We used $\beta = 0.3$ for all DPP node and edge kernel calculations in all simulations.

F Hyperparameters for Real Datasets

Besides the hyperparameters and setup we proposed in Section 6.1 on the synthetic dataset, we report the hyperparameters used for the results on the `MNIST` and `CIFAR10`. As stated in Section 6.2, we used that exact same experiment setup (network architectures, error thresholds, etc.) as in [29] for fair and consistent comparisons. We used SGD optimizers, a learning rate of 0.001, and a momentum of 0.9 for traning on both datasets. For `MNIST`, the training batch size was 1000. For `CIFAR10`, the training batch size was 128. All pruning methods were performed 10 times, and we report the means and standard deviations in Figure 3 (with reweighting) and Figure 7 (without reweighting).

The node-to-edge ratio for pruning, which keeps the number of parameters in the pruned network the same, is $[397 : 614, 472 : 921, 548 : 1228, 623 : 1536, 699 : 1843, 774 : 2150, 849 : 2457, 925 : 2764]$ for `CIFAR10` and $[256 : 156, 287 : 235, 317 : 313, 348 : 392, 378 : 470, 409 : 548, 439 : 627, 470 : 705]$ for `MNIST`, given the network architecture in Table 1 of [29]. These ratios correspond to 20% to 90% of the edges left for each node, as shown on the x-axis of Figure 3 and Figure 7. These node-to-edge ratios are calculated based on the conversion equation in Section 6.2. We used $\beta = 10/|T|$ where $|T|$ is the size of the training dataset for all DPP node and edge kernel calculations on real data, following the choice of [29].

G Tables and Figures

Table 3 shows the experimental results on the synthetic data with the setup discussed in Section 6.1. For all the node-to-edge ratios in (4), given $K = 6$ and $M = 2$, we calculated the mean square GEs for both the noiseless and noisy case ($\sigma = 0.25$). We sampled 100 masks per simulation, and there are in total 10 rounds of simulations. As mentioned in Section 6.1, DPP methods are stable, and the standard deviations are in the magnitude of 10^{-3} for all ratios.

We observe that, DPP edge pruning in the noisy case with 83% of the parameters outperforms the original unpruned loss in simulations (Table 3). Similar behaviour for DPP edge pruning is also observed for real data (see Figure 6.2) when reweighting is applied. We relate this loosely to the *lottery ticket hypothesis*. However, we clearly acknowledge the fact that our pruned model is not reinitialized and retrained, thereby deviating from the original hypothesis.

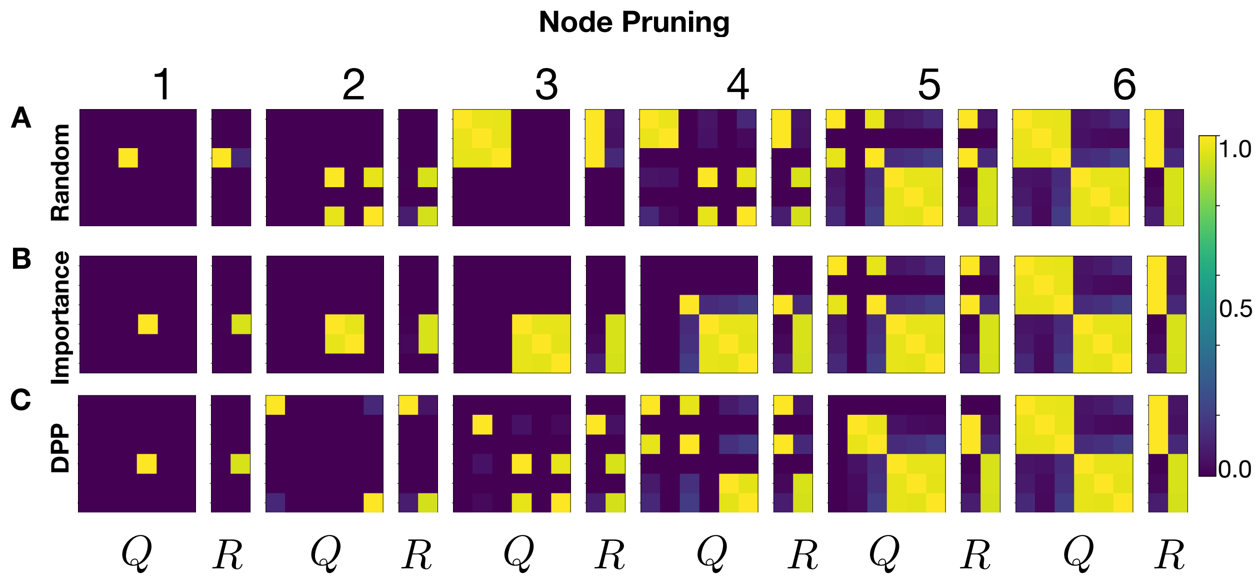

Figure 5: Order parameters after different node pruning methods in the teacher student setup. For this example, number of student hidden nodes $M = 2$ and number of teacher hidden nodes $K = 6$. From [29] we know that the first 3 student nodes (call 1^{st} group) learn one teacher node, and the next 3 (call 2^{nd} group) learn the second teacher node. Recall that k_n is the number of student hidden nodes survived after node pruning. In this figure each row represents a particular node pruning method and each column (Q, R) shows results for different choices of k_n (left to right goes from most pruned to unpruned network). **(A)** In case of random node pruning when $k_n = 2$, two student node survives from the 2^{nd} group after pruning. As a result, information about the 1^{st} teacher node is completely lost in the pruned network. **(B)** Importance pruning keeps a student hidden node depending on its outgoing edge weights. The outgoing edge weights of each group is almost equally distributed among themselves, and they sum up to the second layer weight of corresponding teacher node (see Lemma 4). As all the group size is equal (3 for this example), importance node pruning first selects node from the group whose corresponding second layer teacher weight is highest. In our example, it is the second group and hence for $k_n = 1, 2, 3$, it selects node from the second group. Once a group is exhausted, it then selects from another group according to the aforementioned policy and so on. **(C)** For DPP node pruning when $k_n = 2$, two student hidden nodes are chosen from different groups which preserve information about both the teacher nodes. It can also be shown that, in case of node pruning, if at least one representative from a group survives after pruning, then the reweighting can recover the complete information about that block. Hence, in teacher student framework DPP node pruning performs the best among the node pruning methods especially after reweighting.

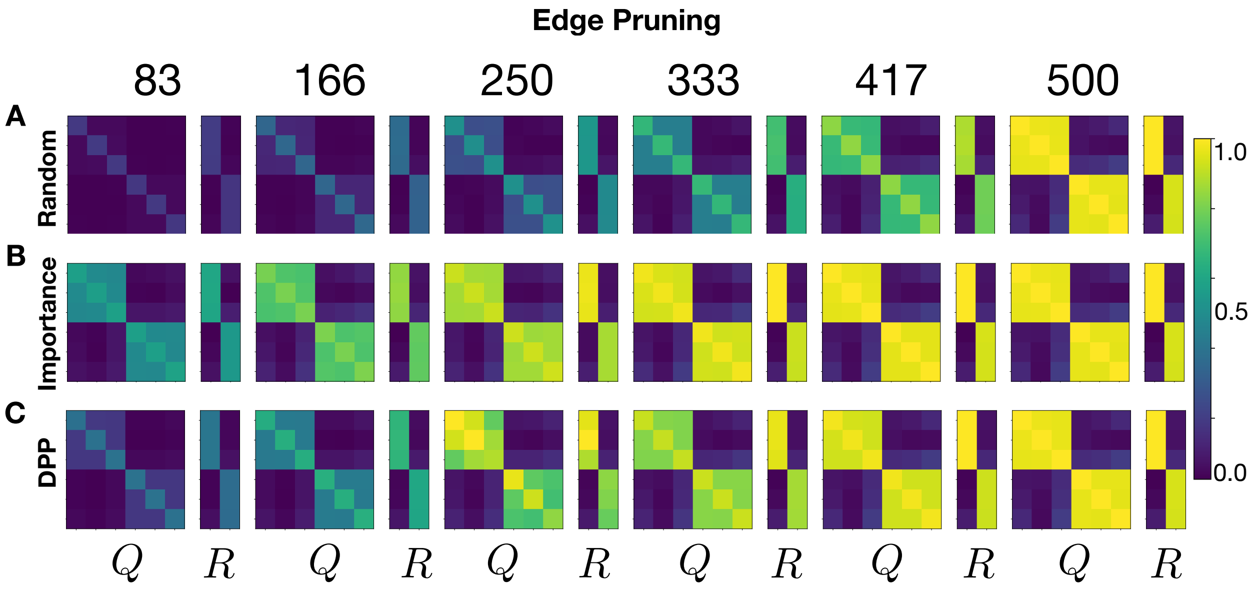

Figure 6: Order parameters after different edge pruning methods in the teacher student setup. For this example, number of student hidden nodes $M = 2$ and number of teacher hidden nodes $K = 6$. From [29] we know that the first 3 student nodes (call 1^{st} group) learn one teacher node, and the next 3 (call 2^{nd} group) learn the second teacher node. Recall that k_e is the number of incoming edges for each student hidden nodes survived after edge pruning. In this figure each row represents a particular edge pruning method and each column (Q, R) shows results for different choices of k_e (left to right goes from most pruned to unpruned network). **(A)** In case of random edge pruning, the expected order parameters have the form described in Lemma 5. **(B)** and **(C)** Order parameters for importance and DPP edge pruning. For importance edge pruning, the edges with lowest absolute values are removed. As the input dimension goes to infinity, the order parameters of the pruned network are close to that of the unpruned network ($k_e = 500$). In case of the DPP edge pruning, the edges are chosen independently w.p. proportional to the absolute value of their weights. However, due to randomness, the order parameters of DPP edge pruning are not that close to the unpruned network's order parameters. In particular, for any fix k_e, let $Q_{k_e}^{imp}$ be the order parameter of the pruned network when importance pruning is used. $Q_{k_e}^{rand}$ and $Q_{k_e}^{DPP}$ are defined similarly. Our simulations show that, $\left\|Q^{unpruned} - Q_{k_e}^{imp}\right\| \leq \left\|Q^{unpruned} - Q_{k_e}^{rand}\right\|$ and $\left\|Q^{unpruned} - Q_{k_e}^{imp}\right\| \leq \left\|Q^{unpruned} - Q_{k_e}^{DPP}\right\|$. This is why the blocks in the Q matrix are the brightest in case of importance pruning. Hence, importance edge pruning performs the best without reweighting, which is also consistent with the performance in real data (Figure 7). However, for real data, after reweighting, dpp edge pruning can recover enough amount of lost information that allows it to outperform all the other pruning methods discussed.

Table 3: The mean square GE on synthetic data for all pruning methods. The left-most row indicates the percentage of parameters left in the network. For specific node-to-edge ratio, see 4. The upper table shows the noiseless case, and the lower shows the noisy case ($\sigma = 0.25$). We also observed the implicit regularization effects of pruning proposed by [29], where DPP Edge in the noisy case with 83% of the parameters outperforms the original unpruned loss.

% of Parameters	DPP Edge	DPP Node	Rand. Edge	Rand. Node	Imp. Edge	Imp. Node
17.0%	2.302± 0.007	3.737± 0.009	3.451± 0.011	3.978 ± 0.016	1.911	3.760
33.0%	1.131± 0.005	2.310± 0.012	2.300± 0.015	2.800 ± 0.035	0.814	2.719
50.0%	0.491± 0.004	1.438± 0.015	1.402± 0.006	1.748 ± 0.036	0.311	1.540
67.0%	0.240± 0.001	0.740± 0.017	0.730± 0.006	1.046 ± 0.018	0.110	0.721
83.0%	0.071± 0.000	0.258± 0.008	0.204± 0.005	0.540 ± 0.010	0.040	0.360
Original Test Loss: 0.051 (Noiseless)						
17.0%	2.398± 0.004	4.000± 0.005	3.769± 0.012	4.188 ± 0.001	1.963	4.167
33.0%	1.213± 0.007	2.622± 0.015	2.558± 0.011	3.041 ± 0.024	0.905	2.910
50.0%	0.608± 0.002	1.633± 0.002	1.675± 0.010	2.023 ± 0.035	0.450	2.031
67.0%	0.300± 0.001	0.890± 0.018	1.007± 0.007	1.269 ± 0.022	0.271	1.144
83.0%	0.159± 0.001	0.394± 0.001	0.490± 0.003	0.643 ± 0.002	0.253	0.659
Original Test Loss: 0.241 ($\sigma = 0.25$)						

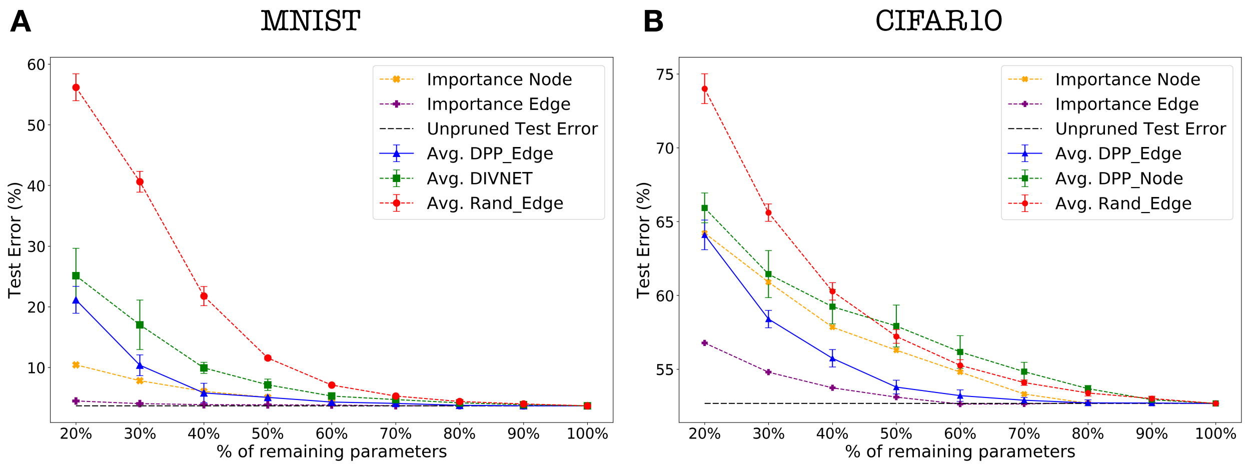

Figure 7: **(A-B)** Comparing different pruning methods on `MNIST` (A) and `CIFAR10` (B) dataset without reweighting. The horizontal axis represents the percentage of remaining parameters in 1^{st} layer after pruning. The vertical axis shows corresponding test error. It can be seen that when reweighting is not applied, importance edge pruning performs the best in both the datasets. This result is consistent with the teacher student framework. However, after reweighting, DPP edge pruning beats all the pruning methods considered. This happens because, in the case of importance pruning, the greedy process of choosing the edges does not keep much room for improvement, whereas choosing the edges using DPP keeps enough space for improvement after reweighting. This shows the novelty of both DPP methods as well as the reweighting procedure.